Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method of real-time shadow generation in computer graphical representation of a scene, the method comprising:

defining an eye's frustum based on a desired view of the scene; defining a location of a light source illuminating at least a portion of the scene;

generating a trapezoid to approximate an area, *E*, within the eye's frustum in a post-perspective space, *L*, of [[a]]the light source[[, *L*]];

applying a trapezoidal transformation to objects within the trapezoid into a trapezoidal space for computing a shadow map; and

determining whether an object or part thereof is in shadow in the desired view of the scene utilising the computed shadow map,

wherein generating the trapezoid comprises:

computing a centre line passing from a position of the eye through E;

calculating a 2D convex hull of E; and

calculating a top line l_t and a base line l_b of the trapezoid using as constraints that l_t and l_b each are orthogonal to the centre line and each touch the 2D convex hull.

2. (Currently Amended) The method as claimed in claim 1, wherein ~~generating the trapezoid comprises generating top and base lines l_t and l_b respectively, of the trapezoid to approximate *E* in *L*, and computing a~~ the centre line ~~l, which~~ passes through centres of near and far planes of *E*~~; calculating a 2D convex hull of *E*; calculating a l_t that is orthogonal to *l* and touches the boundary of the convex hull of *E*; and calculating a l_b which is parallel to l_t and touches the boundary of the convex hull of *E*~~.

3. (Cancelled)

4. (Currently Amended) The method as claimed in claim 1, wherein generating the side lines of the trapezoid to approximate E in L comprises

assigning a distance δ from [[the]]a near plane of the eye's frustum to define a focus region in a desired view of the scene;

determining a point p_L in L that lies on l at the distance δ from the near plane of the eye's frustum;

computing the position of a point q on l, wherein q is the centre of a projection to map the base line and the top line of the trapezoid to y=-1 and y=+1 respectively, and to map p_L to a point on y= ξ, with ξ between -1 and +1; and

constructing two side lines of the trapezoid each passing through q, wherein each sideline touches the 2D convex hull of E on respective sides of l.

5. (Original) The method as claimed in claim 4, wherein ξ =-0.6.

6. (Original) The method as claimed in claim 4, wherein the desired point ξ is determined based on an iterative process that minimizes wastage.

7. (Original) The method as claimed in claim 6, wherein the iterative process is stopped when a local minimum is found.

8. (Previously Presented) The method as claimed in claim 6, wherein the iterative process is pre-computed and the results are stored in a table for direct reference.

9. (Currently Amended) The method as claimed in claim 1, wherein generating the trapezoid further ~~comprising~~ comprises determining an intersection I, between a frustum of the light ~~source's frustum~~ source and the eye's frustum; and

computing the centre point e of the vertices of I;

wherein ~~defining a~~ the centre line ~~l_n passing through~~ passes from the position of the eye and through e~~, for generating the trapezoid~~.

10. (Currently Amended) The method as claimed in claim 9, further comprising defining a new focus region which lies between [[the]] near and far planes of the eye's frustum that are geometrically pushed closer to tightly bound I.

11. (Previously presented) The method as claimed in claim 1, wherein the trapezoidal transformation comprises mapping the four corners of the trapezoid to a unit square that is the shape of a square shadow map, or to a general rectangle that is the shape of a rectangular shadow map.

12. (Original) The method as claimed in claim 11, wherein the size of the square or general rectangle changes based on a configuration of the light source and the eye.

13. (Previously presented) The method as claimed in claim 1, wherein the trapezoidal transformation transforms only the x and the y values of a vertex from the post-perspective space of the light to the trapezoidal space, while the z value is maintained at the value in the post-perspective space of the light.

14. (Previously Presented) The method as claimed in claim 13, further comprising applying the trapezoidal transformation to obtain the x, y, and w values in the trapezoidal space, x_T, y_T, and w_T, and computing the z value in the trapezoidal space, z_T, as $Z_T = \frac{Z_L W_T}{W_L}$, where z_L and w_L, are the z and w values in the post-perspective space of the light, respectively.

15. (Previously Presented) The method as claimed in claim 13, further comprising:

in a first pass of shadow map generation, transforming coordinate values of a fragment from the trapezoidal space back into the post-perspective space *L* of the light to obtain a first transformed fragment, utilising the plane equation of the first transformed fragment to compute a distance value of the first transformed fragment from the light source in *L*, z_{L1}, adding an offset value to z_{L1}, and store the resulting value as a depth value in the shadow map; and

in a second pass of shadow determination, transforming texture coordinate assigned, through projective texturing, to the fragment from the trapezoidal space back into *L*, obtaining a second transformed fragment from the transformed texture coordinate, utilising the plane equation of the second transformed fragment to compute a distance value of the second transformed fragment from the light source in *L*, z_{L2}, and determine whether the fragment is in shadow based on a comparison of the stored depth value in the shadow map and z_{L2}.

16. (Previously Presented) The method as claimed in claims 13, further comprising:

in a first pass of shadow map generation, during a vertex stage, transforming coordinate values of the vertex into the trapezoidal space, and assigning to the vertex the texture coordinate equal to the vertex's coordinate values in the post-perspective space of the light, and during a fragment stage, replacing the depth of the fragment with the texture coordinate of the fragment, adding to the depth an offset, and store the resulting value as a depth value in the shadow map; and

in a second pass of shadow determination, during the vertex stage, transforming coordinate values of the vertex into the post-perspective space of the eye, and assigning to the vertex two texture coordinates that are first the coordinate values of the vertex in the post-perspective space of the light and second the coordinate values of the vertex in the trapezoidal space, and during the fragment stage, determining shadow of the fragment based on a comparison of the stored depth value in the shadow map, as indexed based on the second texture coordinate of the fragment, with a value based on the first texture coordinate of the fragment.

17. (Previously Presented) The method as claimed in claim 13, further comprising:

in a first pass of shadow map generation, transforming coordinate values of a fragment from the trapezoidal space back into the post-perspective space *L* of the light to obtain a first transformed fragment, utilising the plane equation of the first transformed fragment to compute a distance value of the first transformed fragment from the light source in L, z_{L1}, adding an offset value to z_{L1}, and store the resulting value as a depth value in the shadow map; and

in a second pass of shadow determination, during the vertex stage, transforming coordinate values of the vertex into the post-perspective space of the eye, and assigning to the vertex two texture coordinates that are first the coordinate values of the vertex in the post-perspective space of the light and second the coordinate values of the vertex in the trapezoidal space, and during the fragment stage, determining shadow of the fragment based on a comparison of the stored depth value in the shadow map, as indexed based on the second texture coordinate of the fragment, with a value based on the first texture coordinate of the fragment.

18. (Previously Presented) The method as claimed in claim 13, further comprising:

in a first pass of shadow map generation, during a vertex stage, transforming coordinate values of the vertex into the trapezoidal space, and assigning to the vertex the texture coordinate equal to the vertex's coordinate values in the post-perspective space of the light, and during a fragment stage, replacing the depth of the fragment with the texture coordinate of the fragment, adding to the depth an offset, and store the resulting value as a depth value in the shadow map; and

in a second pass of shadow determination, transforming texture coordinate assigned, through projective texturing, to the fragment from the trapezoidal space back into *L*, obtaining a second transformed fragment from the transformed texture coordinate, utilising the plane equation of the second transformed fragment to compute a distance value of the second transformed fragment from the light source in *L*, z_{L2}, and determine whether the fragment is in shadow based on a comparison of the stored depth value in the shadow map and z_{L2}.

19. (Previously presented) The method as claimed in claim 1, further comprising adding a polygon offset in the determining whether an object or part thereof is in shadow in the desired view of the scene for representation utilising the computed shadow map.

20. (Previously presented) The method as claimed in claim 1, wherein two or more light sources illuminate at least respective portions of the scene, and the method is applied for each light source.

21. (Currently Amended) A system for real-time shadow generation in computer graphical representation of a scene, the system comprising: a processor unit for defining an eye's frustum based on a desired view of the scene; for defining a location of a light source illuminating at least a portion of the scene; for generating a trapezoid to approximate an area, *E*, within the eye's frustum in the post-perspective space, *L*, of the light~~, *L*, from the light~~ source; for applying a trapezoidal transformation to objects within the trapezoid into a trapezoidal space, for computing a shadow map; and for determining whether an object or part thereof is in shadow in the desired view of the scene utilising the computed shadow map;

wherein the processor unit, in generating the trapezoid, computes a centre line passing from a position of the eye through E; calculates a 2D convex hull of E; and calculates a top line l_t and a base line l_b of the trapezoid using as constraints that l_t and l_b each are orthogonal to the centre line and each touch the 2D convex hull.

22. (Currently Amended) A data storage medium having stored thereon computer code means for instructing a computer to execute a method of real-time shadow generation in computer graphical representation of a scene, the method comprising: defining an eye's frustum based on a desired view of the scene; defining a location of a light source illuminating at least a portion of the scene; generating a trapezoid to approximate an area, *E*, within the eye's frustum in the post-perspective space, *L*, of the light~~, *L*, from the light~~ source; applying a trapezoidal transformation to objects within the trapezoid into a trapezoidal space for computing a shadow

map; and determining whether an object or part thereof is in shadow in the desired view of the scene utilising the computed shadow map;

wherein generating the trapezoid comprises:

computing a centre line passing from a position of the eye through E;

calculating a 2D convex hull of E; and

calculating a top line l_t and a base line l_b of the trapezoid using as constraints that l_t and l_b each are orthogonal to the centre line and each touch the 2D convex hull.